A Repository of Jupyter Notebooks on Unlearning in Federated Learning

First Presentation

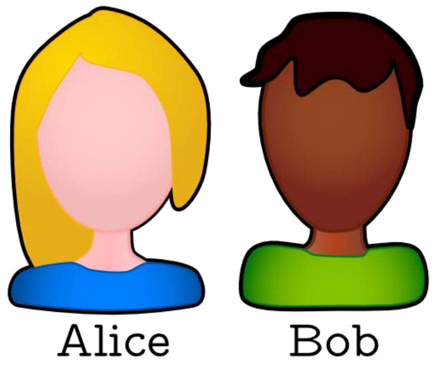
FYP22002 @ HKUCS

Sheng “Victor” HUANG

Supervisor: Prof. S M YIU

18 Jan 2023

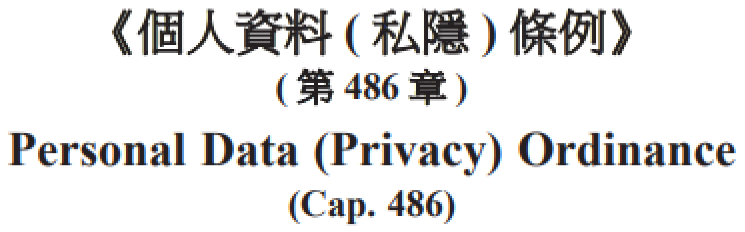
Introduction



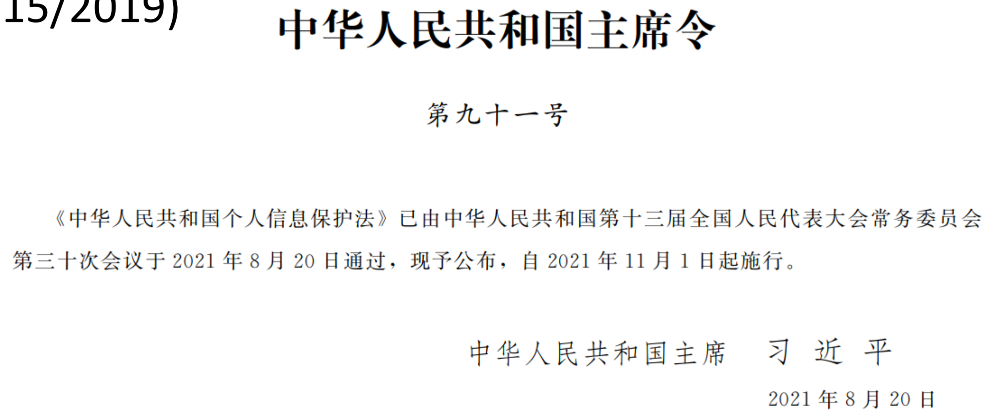
- A Story
  - Alice lent her laptop to Bob yesterday
  - Bob googled “heart disease”
  - Bob wiped his search history and returned the laptop to Alice today
  - Alice finds heart disease medicine on various sites that use Google Ads
  - Now Alice thinks Bob might have a heart disease
  - Google thinks Alice might have a heart disease
- Why?
  - Search history used to train ML models in Ad recommendation system
  - Wiping search history $\nLeftrightarrow$ wiping data influence in ML models
- Need to erase data from ML models

Introduction

- Privacy – legal requirement
  - Right to be Forgotten (RtbF)


《個人資料(私隱)條例》
(第486章)
Personal Data (Privacy) Ordinance
(Cap. 486)

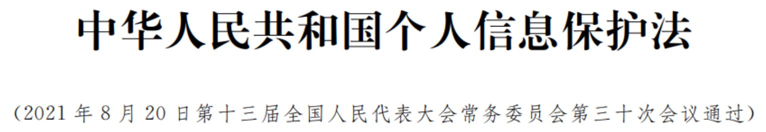
Note 1:
No standalone RtbF in HKSAR, as there is no such right mentioned in the PDPO. Data erasure is pursuant to Data Protection Principle 2 and section 26 of the PDPO.
(2020 Administrative Appeals Board Decision -> Appeal No. 15/2019)


中华人民共和国主席令

第九十一号

《中华人民共和国个人信息保护法》已由中华人民共和国第十三届全国人民代表大会常务委员会第三十次会议于 2021 年 8 月 20 日通过，现予公布，自 2021 年 11 月 1 日起施行。

中华人民共和国主席　习　近　平
2021 年 8 月 20 日


中华人民共和国个人信息保护法

(2021 年 8 月 20 日第十三届全国人民代表大会常务委员会第三十次会议通过)

Note 2:
No RtbF mentioned in PRC’s Personal Information Protection Law (PIPL).

Introduction

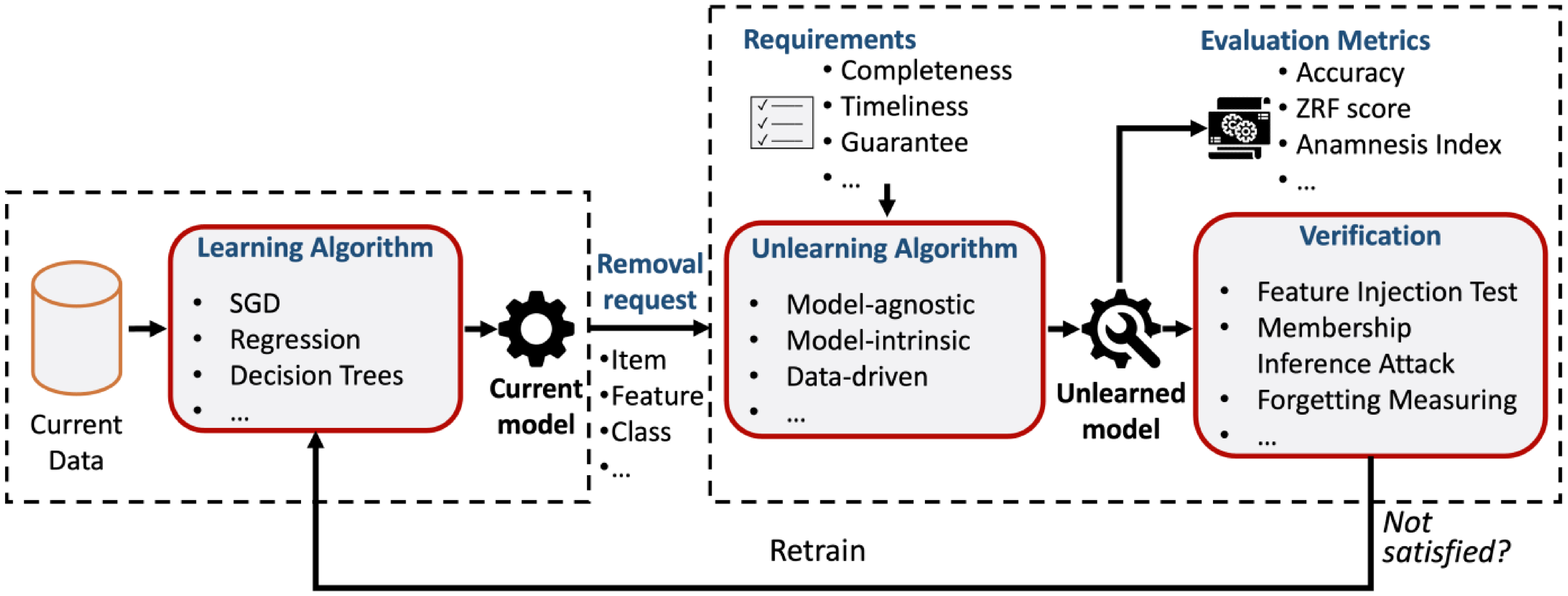
- Not just privacy
  - Security: data leakage, adversarial attacks (model stealing, membership inference, poisoning), …
  - Usability: outdated/incorrect data in ML systems
  - Fidelity: human bias (biased data on gender, age, race, …)

Machine Unlearning [CY15]

- Remove data (and its influence) from ML models
- ML models as black boxes [KL17]
- Naïve method -> to retrain on remaining data -> slow
- Challenges:
  - Stochasticity of training
    - Data influence difficult to track
  - Incrementality of training
    - Training with a data point affected by prior training, and will affect later training
  - Catastrophic unlearning
    - Unlearnt models generally perform worse than retrained models

[CY15] https://ieeexplore.ieee.org/document/7163042
[KL17] https://proceedings.mlr.press/v70/koh17a

Unlearning Framework



[Ngu+22] https://arxiv.org/abs/2209.02299

Untraining Framework

- Unlearning requests
  - Item removal
  - Feature removal
  - Class removal
  - Task removal
  - Stream removal
  - …
- Design requirements
  - Completeness (consistency)
  - Timeliness
  - Accuracy
  - Light-weight
  - Provable guarantees
  - Model-agnostic
  - Verifiability
  - …
- Unlearning verification
  - Membership inference attacks
  - Backdoor attacks
  - Slow-down attack
  - Feature injection
  - Forgetting measure
  - Information leakage
  - Cryptographic protocol
  - …

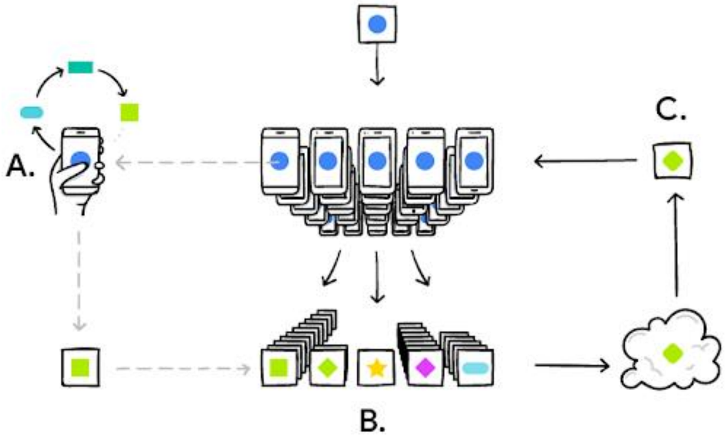
[Ngu+22] https://arxiv.org/abs/2209.02299

Unlearning Methods

- Exact unlearning
  - Only guaranteed with retraining or variations of retraining
  - Slow, especially with complex models
  - Variations usually take snapshots during training [Bou+21, Ngu+22]
- Approximate unlearning
  - Fisher unlearning [Mar20]
    - Uses the **remaining data**, takes a corrective Newton step and injects optimal noise
  - Influence unlearning
    - Computes the influence of the **target data** and remove it from model
  - Gradient unlearning
    - Approximates SGD steps as if retraining was done, using **remaining data**

[Bou+21] https://ieeexplore.ieee.org/document/9519428
[Ngu+22] https://arxiv.org/abs/2209.02299
[Mar20] https://dl.acm.org/doi/abs/10.5555/3455716.3455862

Federated Learning (FL) [McM+17]

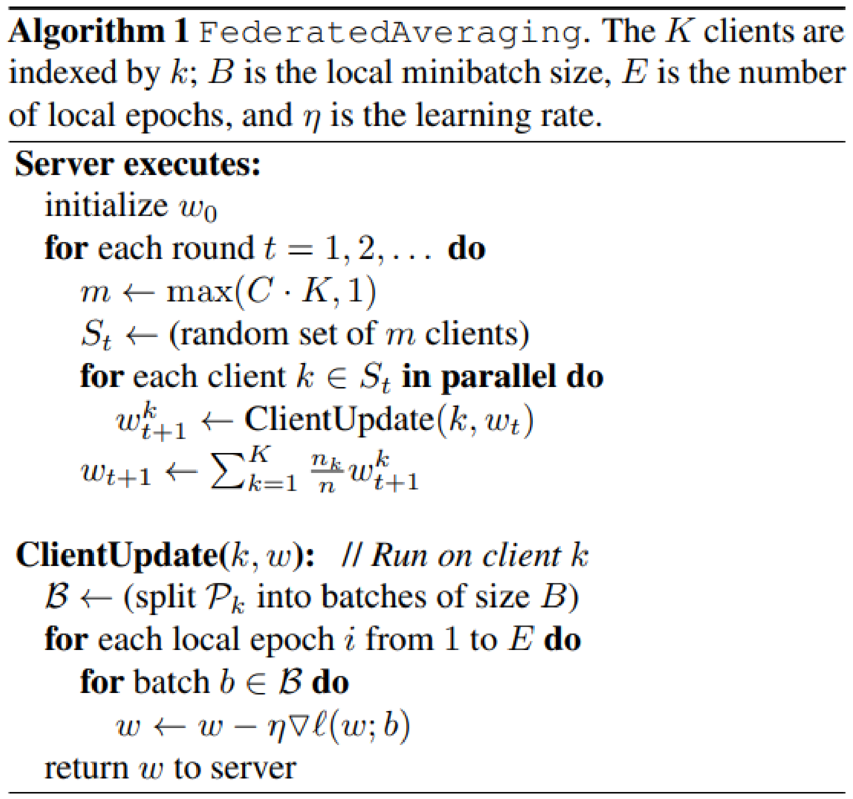


- Central server S initializes a model M_0;
- For training in iteration i, Central server S sends M_i to a group of K decentralized clients $C = \{C_1, C_2, \ldots, C_K\}$;
- Each client $C_k \in C$ trains the model with its own data, generating an update U_k^i;
- Clients send updates $\{U_1^i, U_2^i, \ldots, U_K^i\}$ to S;
- S averages all the updates $\{U_1^i, U_2^i, \ldots, U_K^i\}$ to U^i and updates the model to $M_{i+1} = M_i + U^i$;
- When the termination criteria are met, S stop training and output the model M_{i+1} as M;
- If not terminated, S sends M_{i+1} to C for iteration $i + 1$.

[McM+17] https://proceedings.mlr.press/v54/mcmahan17a.html

Federated Learning (FL) [McM+17]

- FedAvg – most important algorithm in FL



Algorithm 1 `FederatedAveraging`. The K clients are indexed by k; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.

```
Server executes:
  initialize $w_0$
  for each round $t = 1, 2, \dots$ do
    $m \leftarrow \max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of $m$ clients)
    for each client $k \in S_t$ in parallel do
      $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate($k, w$):   // Run on client $k$
  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
  for each local epoch $i$ from 1 to $E$ do
    for batch $b \in \mathcal{B}$ do
      $w \leftarrow w - \eta \nabla \ell(w; b)$
  return $w$ to server
```

[McM+17] https://proceedings.mlr.press/v54/mcmahan17a.html

Federated Unlearning

Remember challenges for centralized unlearning:

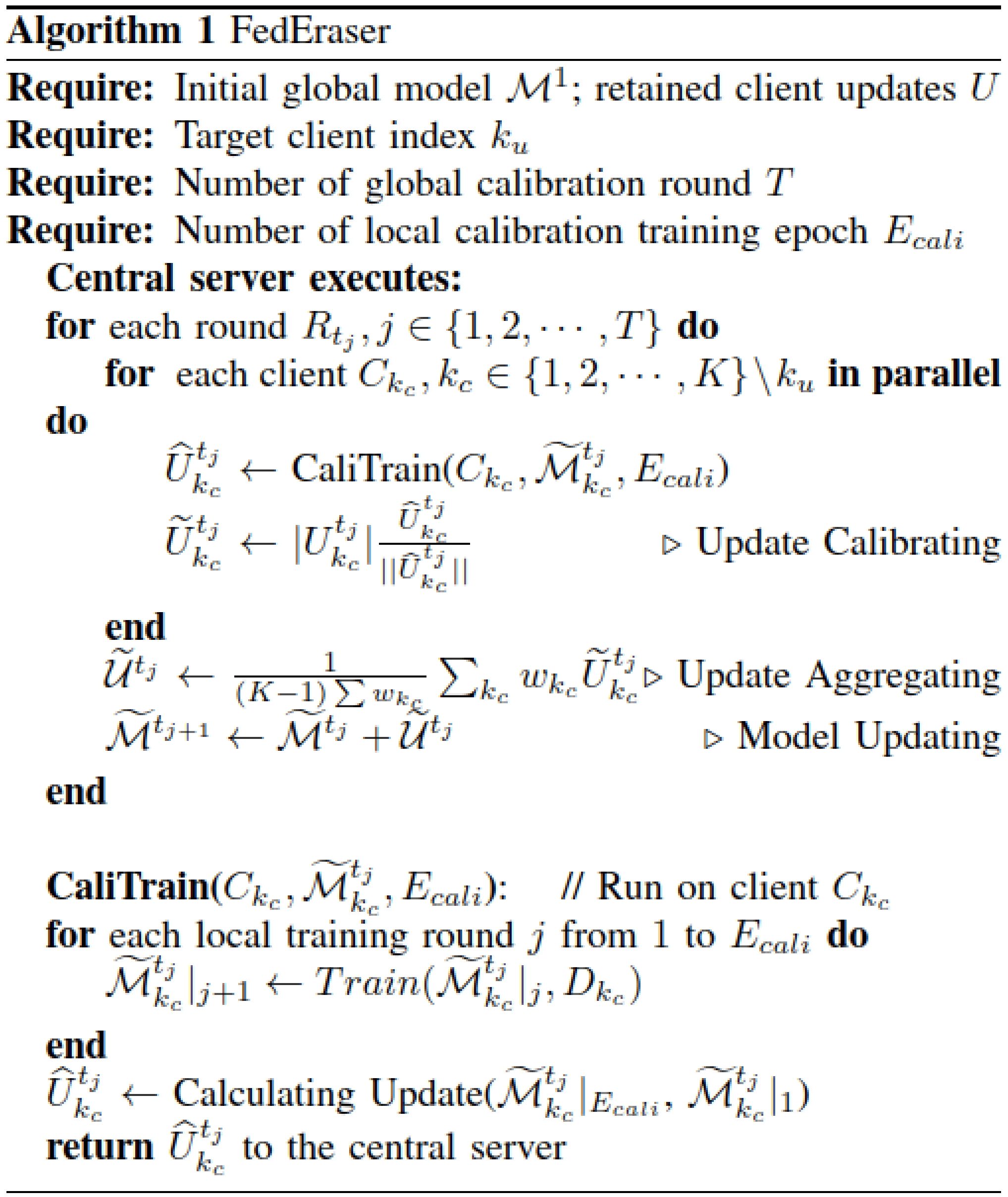
- Stochasticity of training
  - Data influence difficult to track
- Incrementality of training
  - Training with a data point affected by prior training, and will affect later training
- Catastrophic unlearning
  - Unlearnt models generally perform worse than retrained models

Additional Challenges

- Limited data access
  - Server doesn't have access to training data used at the client side
  - This makes unlearning methods for centralized ML difficult to implement for FL
- Limited client participation
  - Client goes offline and can't participate in unlearning process
- Complicated relationship between data and clients
  - Unlearning all data of one client of just part of it?
  - What if there is data overlap on other clients?
- Data partition
  - Horizontal FL, vertical FL and federated transfer learning
- Statistical heterogeneity
  - IID (Independent and Identically Distributed)
- Adversarial model
  - Untruthful server/clients
- …

FedEraser [Liu+21a]

- First attempt
- For each remaining client
  - Calibration (local) training with its own data
  - Sends the new update to the server
- For the server
  - Calibrate saved updates
  - Aggregate the calibrated updates
  - Update the model



Algorithm 1 FedEraser

```
Require: Initial global model $\mathcal{M}^1$; retained client updates $U$
Require: Target client index $k_u$
Require: Number of global calibration round $T$
Require: Number of local calibration training epoch $E_{cali}$
  Central server executes:
  for each round $R_{t_j}, j \in \{1, 2, \cdots, T\}$ do
     for each client $C_{k_c}, k_c \in \{1, 2, \cdots, K\} \setminus k_u$ in parallel
  do
        $\widehat{U}^{t_j}_{k_c} \leftarrow$ CaliTrain($C_{k_c}, \widetilde{\mathcal{M}}^{t_j}_{k_c}, E_{cali}$)
        $\widetilde{U}^{t_j}_{k_c} \leftarrow |U^{t_j}_{k_c}| \frac{\widehat{U}^{t_j}_{k_c}}{||\widehat{U}^{t_j}_{k_c}||}$        ▷ Update Calibrating
     end
     $\widetilde{\mathcal{U}}^{t_j} \leftarrow \frac{1}{(K-1)\sum w_{k_c}} \sum_{k_c} w_{k_c} \widetilde{U}^{t_j}_{k_c}$ ▷ Update Aggregating
     $\widetilde{\mathcal{M}}^{t_{j+1}} \leftarrow \widetilde{\mathcal{M}}^{t_j} + \widetilde{\mathcal{U}}^{t_j}$        ▷ Model Updating
  end

  CaliTrain($C_{k_c}, \widetilde{\mathcal{M}}^{t_j}_{k_c}, E_{cali}$):    // Run on client $C_{k_c}$
  for each local training round $j$ from 1 to $E_{cali}$ do
     $\widetilde{\mathcal{M}}^{t_j}_{k_c}|_{j+1} \leftarrow Train(\widetilde{\mathcal{M}}^{t_j}_{k_c}|_j, D_{k_c})$
  end
  $\widehat{U}^{t_j}_{k_c} \leftarrow$ Calculating Update($\widetilde{\mathcal{M}}^{t_j}_{k_c}|_{E_{cali}}, \widetilde{\mathcal{M}}^{t_j}_{k_c}|_1$)
  return $\widehat{U}^{t_j}_{k_c}$ to the central server
```

[Liu+21a] https://ieeexplore.ieee.org/document/9521274

FedEraser [Liu+21a]

In what direction should the model change?

How much should the model change?

$$\widetilde{U}^{t_j}_{k_c} = \boxed{\left|U^{t_j}_{k_c}\right|}\ \boxed{\frac{\widehat{U}^{t_j}_{k_c}}{\left\|\widehat{U}^{t_j}_{k_c}\right\|}}$$

[Liu+21a] https://ieeexplore.ieee.org/document/9521274

FedEraser [Liu+21a]

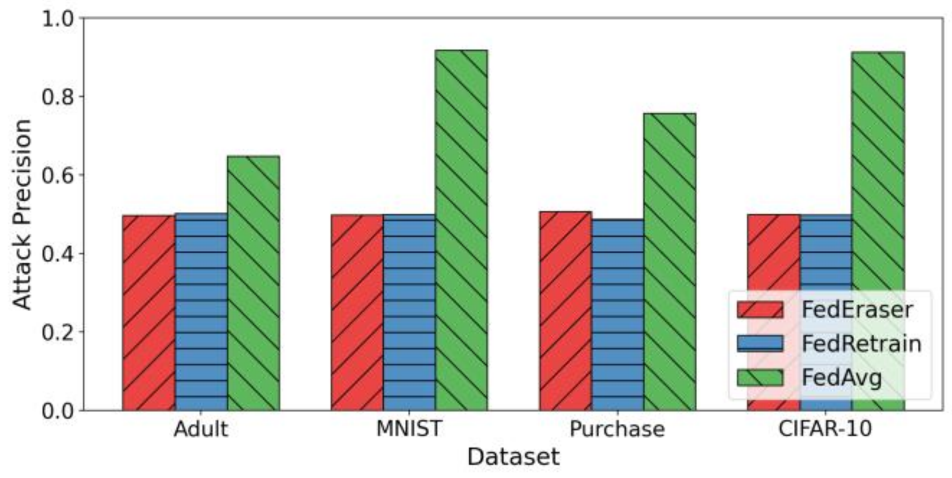
Advantages

- Minor change to FL architecture
- Much faster than retraining

Limitations

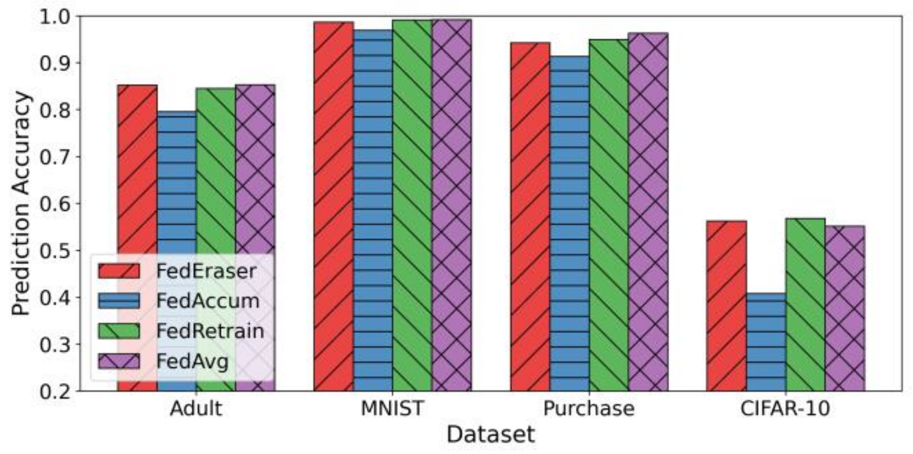
- Require client participation
- Doesn't work well with complex models
- Calibration's effectiveness under-explored

[Liu+21a] https://ieeexplore.ieee.org/document/9521274



(a) Attack Precision

Membership inference attacks show similar data erasure performance as retraining.



(a) Prediction Accuracy (Testing Data)

While performance doesn't drop too much.

Federated Unlearning with Knowledge Distillation [WZM22]

$$M_F = M_1 + \sum_{t=1}^{F-1} \Delta M_t$$

$$\Delta M_t = \frac{1}{N}\sum_{i=1}^{N} \Delta M_t^i = \frac{1}{N}\sum_{i=1}^{N-1} \Delta M_t^i + \frac{1}{N}\Delta M_t^N$$

$$\Delta M_t' = \frac{1}{N-1}\sum_{i=1}^{N-1} \Delta M_t^i = \frac{N}{N-1}\Delta M_t - \frac{1}{N-1}\Delta M_t^N$$

$$M_F' = M_1 + \frac{N}{N-1}\sum_{t=1}^{F-1} \Delta M_t - \frac{1}{N-1}\sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t$$

$$\Delta M_t' = \frac{1}{N}\sum_{i=1}^{N-1} \Delta M_t^i = \Delta M_t - \frac{1}{N}\Delta M_t^N$$

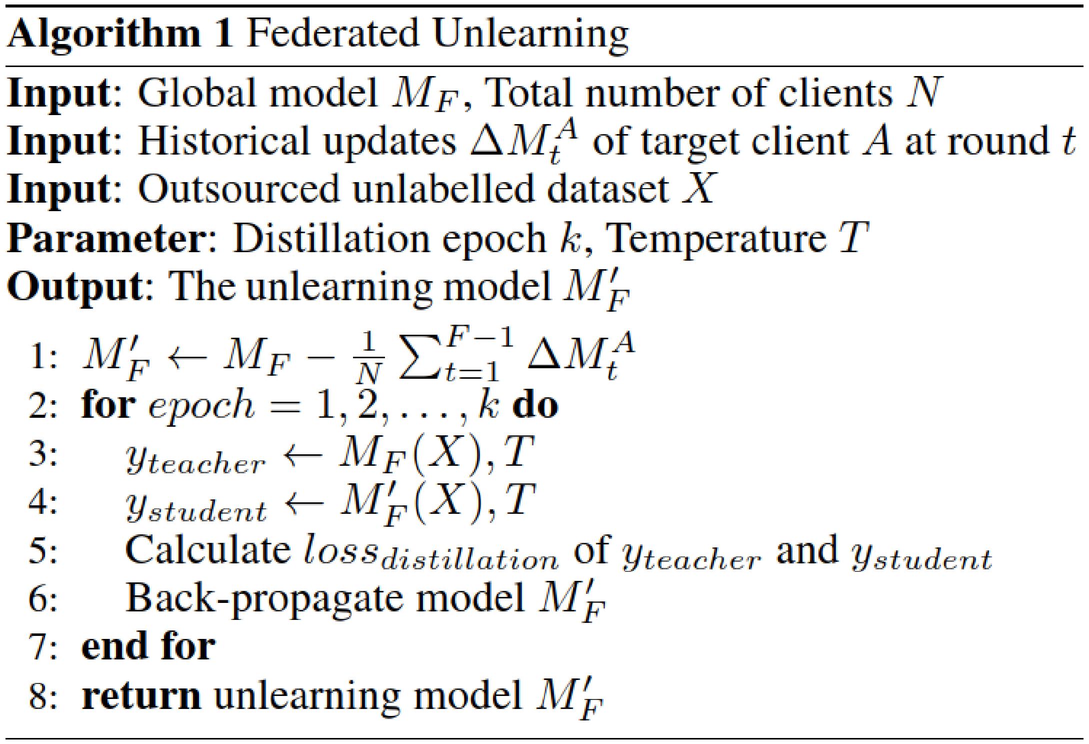
Assume client N participated but generated update is 0

[WZM22] https://arxiv.org/abs/2201.09441

Federated Unlearning with Knowledge Distillation [WZM22]

$$\begin{aligned} M'_F &= M_1 + \sum_{t=1}^{F-1} \Delta M'_t + \sum_{t=1}^{F-1} \epsilon_t \\ &= M_1 + \sum_{t=1}^{F-1} \Delta M_t - \frac{1}{N} \sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t \\ &= M_F - \frac{1}{N} \sum_{t=1}^{F-1} \Delta M_t^N + \sum_{t=1}^{F-1} \epsilon_t \end{aligned}$$

No good method to calculate ϵ_t. Use distillation to estimate.



Algorithm 1 Federated Unlearning

Input: Global model M_F, Total number of clients N
Input: Historical updates ΔM_t^A of target client A at round t
Input: Outsourced unlabelled dataset X
Parameter: Distillation epoch k, Temperature T
Output: The unlearning model M'_F

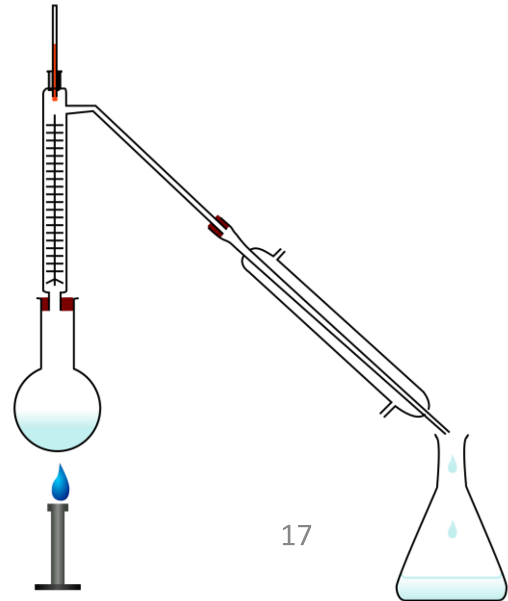
```
1: $M'_F \leftarrow M_F - \frac{1}{N}\sum_{t=1}^{F-1} \Delta M_t^A$
2: for $epoch = 1, 2, \ldots, k$ do
3:     $y_{teacher} \leftarrow M_F(X), T$
4:     $y_{student} \leftarrow M'_F(X), T$
5:     Calculate $loss_{distillation}$ of $y_{teacher}$ and $y_{student}$
6:     Back-propagate model $M'_F$
7: end for
8: return unlearning model $M'_F$
```

[WZM22] https://arxiv.org/abs/2201.09441

Federated Unlearning with Knowledge Distillation [WZM22]

Knowledge Distillation [HVD15]

- Use the prediction results of class probabilities produced by a teacher model to train a student model
- Knowledge acquired by model is not only encoded in weight parameters but also the class probability prediction output.



[WZM22] https://arxiv.org/abs/2201.09441
[HVD15] https://arxiv.org/abs/1503.02531

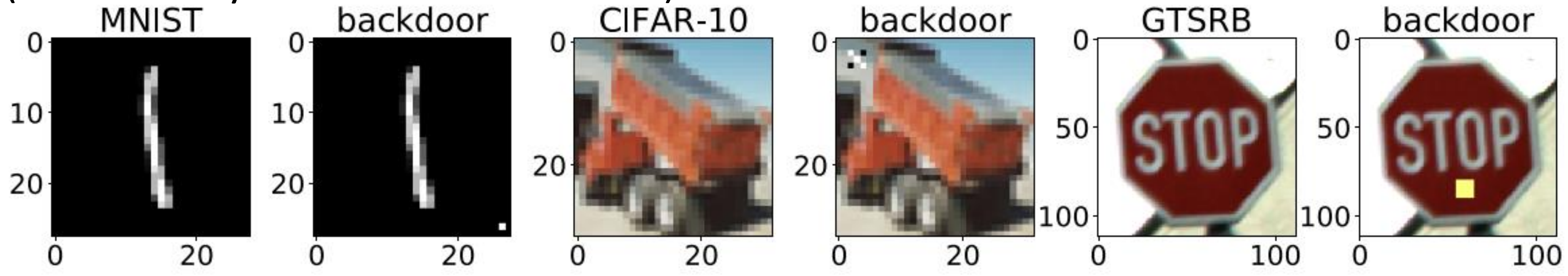
Federated Unlearning with Knowledge Distillation [WZM22]

Advantages

- No client participation needed
- Works well with complex models such as DNNs

Limitations

- Process is costly and approximate
- Original model may still be at client side
- Verification proposed in paper (backdoor attack) is not ideal in real world situations (intentionally introduces a backdoor)



[WZM22] https://arxiv.org/abs/2201.09441

Other Methods

- [Liu+21b] designed a novel framework of revocable (vertical) federated random forest (RevFRF). RevFRF uses a suite of homomorphic encryption (HE) based secure protocols to implement RF, enabling model construction, prediction and participant revocation, even with untruthful server and clients.
- [GSK21] studied federated learning and unlearning in a decentralized network within a Bayesian framework. It developed federated variational inference (VI) solutions based on the decentralized solution of local free energy minimization problems within exponential-family models and on local gossip-driven communication.

[Liu+21b] https://ieeexplore.ieee.org/document/9514457
[GSK21] https://ieeexplore.ieee.org/document/9593225

Other Methods

- [Wan+22] proposed a method for scrubbing information from CNNs in FL when an entire class need to be forgotten (class-level unlearning), using Term Frequency Inverse Document Frequency (TF-IDF) to quantize the class discrimination of channels.
- [Liu+22] proposed a smart retraining method for federated unlearning without communication protocols. The approach uses the L-BFGS algorithm to efficient solve a Hessian approximation with historical parameter updates for global model retraining.

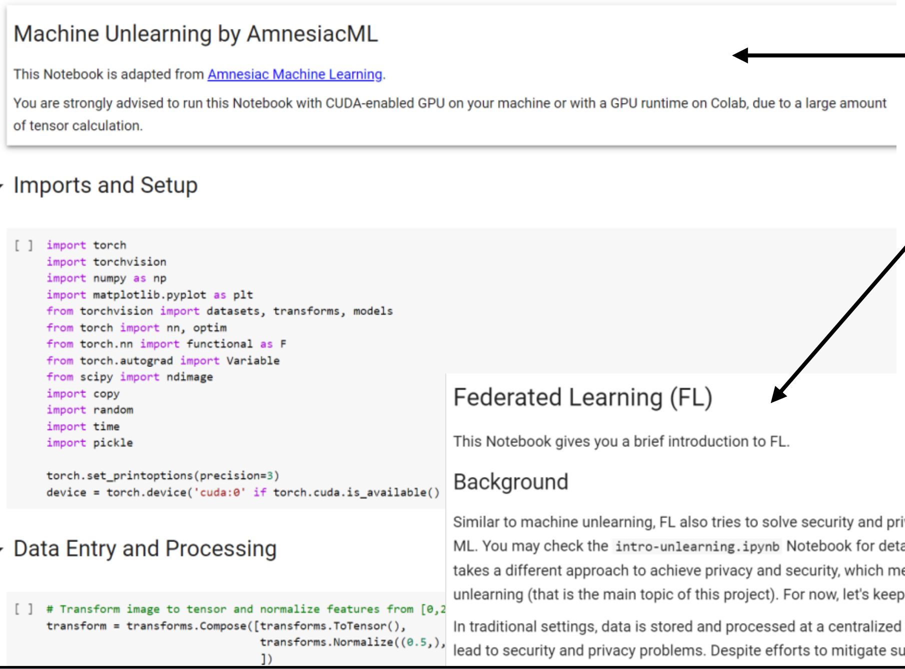
[Wan+22] https://dl.acm.org/doi/abs/10.1145/3485447.3512222
[Liu+22] https://ieeexplore.ieee.org/document/9796721

Comparison

- [Liu+21a] and [WZM22] have more general methods that require little change to the existing FL framework. [Liu+21a] doesn’t work well with complex models, while [WZM22] don’t need client participation (both an advantage and a limitation).
- [Liu+21b], [GSK21] and [Wan+22] all designed unlearning methods that are very specific to unlearning requests, models, FL data partition, learning algorithms, etc.
- [Liu+22] proposed a rapid retraining method. However, it stores snapshots, which can bring burden to storage and creates privacy risk. And it also don’t work well with complex models.

[Liu+21a] https://ieeexplore.ieee.org/document/9521274
[WZM22] https://arxiv.org/abs/2201.09441
[Liu+21b] https://ieeexplore.ieee.org/document/9514457
[GSK21] https://ieeexplore.ieee.org/document/9593225
[Wan+22] https://dl.acm.org/doi/abs/10.1145/3485447.3512222
[Liu+22] https://ieeexplore.ieee.org/document/9796721

Progress



Machine Unlearning by AmnesiacML

This Notebook is adapted from Amnesiac Machine Learning.

You are strongly advised to run this Notebook with CUDA-enabled GPU on your machine or with a GPU runtime on Colab, due to a large amount of tensor calculation.

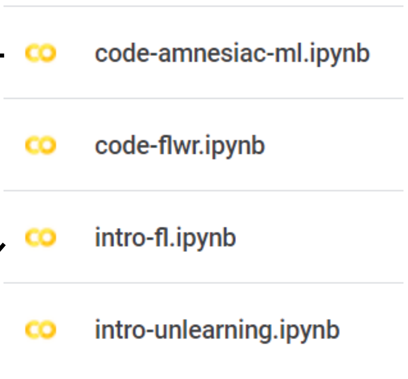
Imports and Setup

```
[ ]  import torch
     import torchvision
     import numpy as np
     import matplotlib.pyplot as plt
     from torchvision import datasets, transforms, models
     from torch import nn, optim
     from torch.nn import functional as F
     from torch.autograd import Variable
     from scipy import ndimage
     import copy
     import random
     import time
     import pickle

     torch.set_printoptions(precision=3)
     device = torch.device('cuda:0' if torch.cuda.is_available()
```

Data Entry and Processing

```
[ ]  # Transform image to tensor and normalize features from [0,2
     transform = transforms.Compose([transforms.ToTensor(),
                                     transforms.Normalize((0.5,),
                                     ])
```

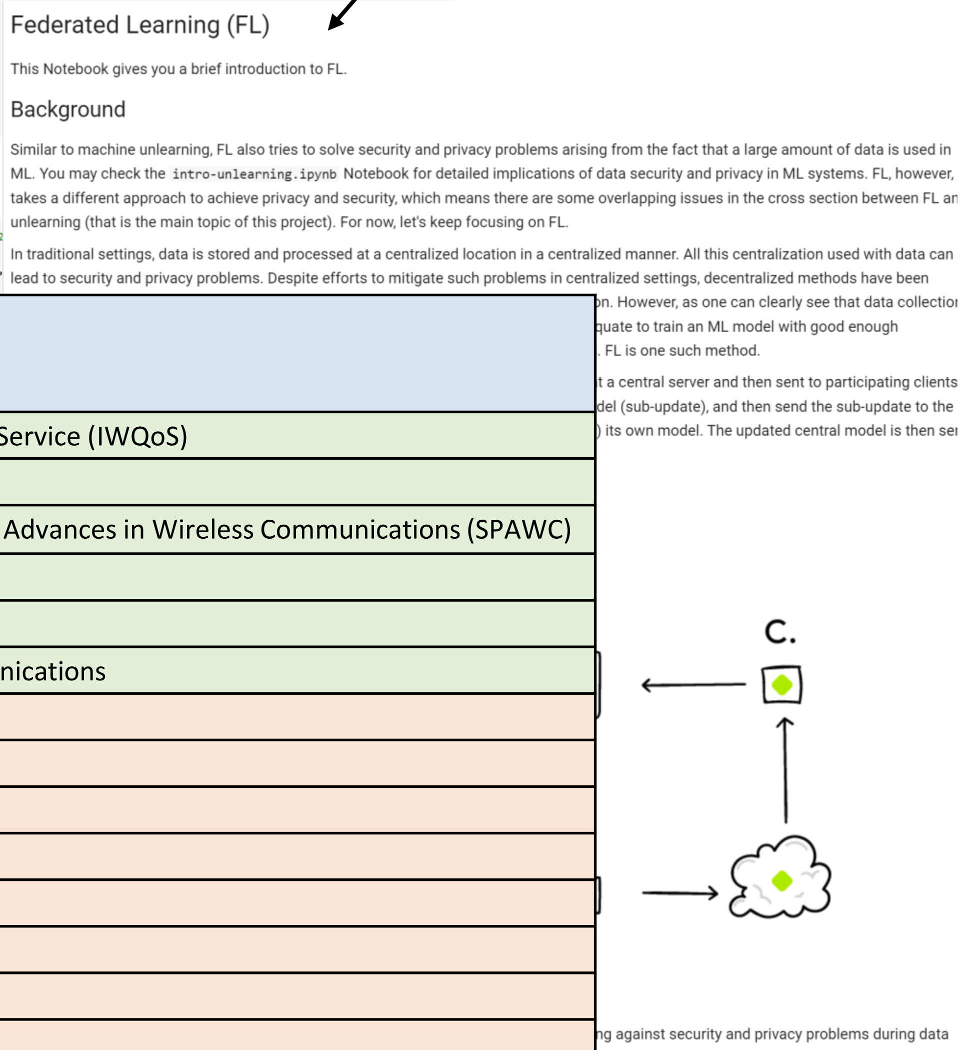


code-amnesiac-ml.ipynb

code-flwr.ipynb

intro-fl.ipynb

intro-unlearning.ipynb



Federated Learning (FL)

This Notebook gives you a brief introduction to FL.

Background

Similar to machine unlearning, FL also tries to solve security and privacy problems arising from the fact that a large amount of data is used in ML. You may check the `intro-unlearning.ipynb` Notebook for detailed implications of data security and privacy in ML systems. FL, however, takes a different approach to achieve privacy and security, which means there are some overlapping issues in the cross section between FL an unlearning (that is the main topic of this project). For now, let's keep focusing on FL.

In traditional settings, data is stored and processed at a centralized location in a centralized manner. All this centralization used with data can lead to security and privacy problems. Despite efforts to mitigate such problems in centralized settings, decentralized methods have been

on. However, as one can clearly see that data collectio

quate to train an ML model with good enough

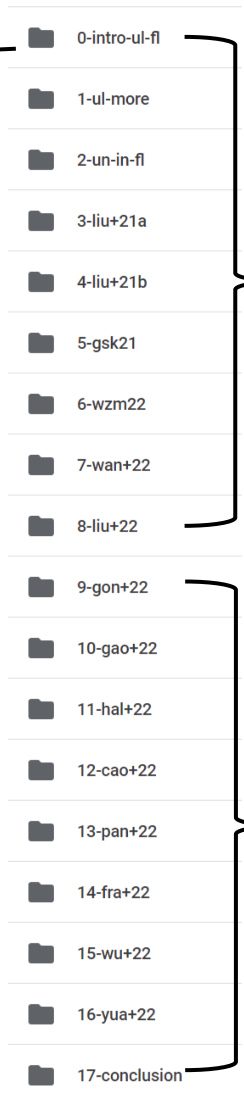
. FL is one such method.

t a central server and then sent to participating clients

del (sub-update), and then send the sub-update to the

) its own model. The updated central model is then se

ng against security and privacy problems during data



0-intro-ul-fl

1-ul-more

2-un-in-fl

3-liu+21a

4-liu+21b

5-gsk21

6-wzm22

7-wan+22

8-liu+22

9-gon+22

10-gao+22

11-hal+22

12-cao+22

13-pan+22

14-fra+22

15-wu+22

16-yua+22

17-conclusion

Done

To do

Paper	Date of Publication / Conference / Preprint Last Edit	Venue
[Liu+21a]	25-28 June 2021	2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQoS)
[Liu+21b]	16 Aug 2021	IEEE Transactions on Dependable and Secure Computing
[GSK21]	27-30 Sep 2021	2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)
[WZM22]	24 Jan 2022	arXiv
[Wan+22]	25 April 2022	Proceedings of the ACM Web Conference 2022 (WWW)
[Liu+22]	02-05 May 2022	IEEE INFOCOM 2022 - IEEE Conference on Computer Communications
[Gon+22]	22-23 May 2022	2022 IEEE Data Science and Learning Workshop (DSLW)
[Gao+22]	25 May 2022	arXiv
[Hal+22]	9 Sep 2022	arXiv
[Cao+22]	20 Oct 2022	arXiv (to appear in S&P 2023 in May)
[Pan+22]	28 Oct 2022	arXiv
[Fra+22]	21 Nov 2022	arXiv
[Wu+22]	25 Nov 2022	arXiv
[Yua+22]	03 Dec 2022	arXiv

Some Takeaways

- FL has many variations. It’s difficult to find an unlearning method that both performs well and fits that many variations.
- FL is still in its early stages and not a practical privacy-enhancing technology (PET) yet [Boe+23], so there will be a lot of federated learning and unlearning research work done in the future.
- Some trends and open questions on unlearning: defining success of unlearning (verification / data auditing); unified unlearning requirements; unified unlearning benchmarking; adversarial machine unlearning; interpretable machine unlearning; …

[Boe+23] https://arxiv.org/abs/2301.04017

Thank you!

- What have you learnt about federated unlearning?
  - I forgot… Maybe ask my clients.

END
完